Implementation and Algorithms for Vertex QT-DSM Tree
RHIC 2011 200GeV AuAu Run

Eleanor Judd
Chris Perkins

May 25, 2011

Change Log:

Date	Description
January 11, 2011	First version for 2011 proton-proton data taking. The ZD101 algorithm has been changed. The front/back bits from 2009 have been restored. Also, the size of the TAC values produced by the QT boards have been reduced from 12 to 10 bits, to make room for truncated sums from each of the East and West sides. The new ZD101 algorithm passes those sums through to the scaler board, instead of passing some of the TAC bits. The ZD101 output to VT201 has been kept the same as in 2009, so the VT201 algorithm from 2009 will be re-used
March 30, 2011	First Version for 2011 19 GeV AuAu data taking. No coding changes to any algorithms. We just reverted to the 2010 algorithms.
May 5, 2011	First change for 2011 200 GeV AuAu data taking. The VT201 algorithm has been changed to include preceded protection for the minimum bias bit.
May 25, 2011	The VT201 algorithm has been changed again. The preceded protection has been removed from the primary minimum bias output bit. The secondary minimum bias bit has been replaced with that preceded protection bit. The combinations can now be done at the TCU level. At the same time, a minor bug in the atomcules logic has been fixed.

The Vertex branch of the DSM tree is used to locate the primary vertex of the RHIC beam collisions at STAR. All three relevant trigger detectors connect to this branch: Zero Degree Calorimeters (ZDC), Beam-Beam Counters (BBC) and the Vertex Position Detector (VPD). The raw detector signals are digitized and pre-processed in QT boards. The DSM tree is then used to calculate TAC differences and combine ADC information to produce (for example) minimum bias, ultra-peripheral or atomcules triggers.

1. Layer 0 QT Boards: BBQ_BB001:002

There are two BBC small-tile QT boards: one processes data from the East side of the detector and the other from the West side. Please see documentation provided by Chris Perkins for a description of this algorithm

2. Layer 1 DSM Boards: BBC_BB101

The BB101 DSM board processes data from the BBC-small-tile detector. The algorithm receives ADC-sum and fastest-TAC data from the QT boards. The ADC sums are compared to thresholds. A set of bits specified by the user is chosen from each incoming TAC value to send to the scaler system. In parallel, the TAC difference is calculated. The difference is set to

zero if either of the two incoming TACS is zero, because a TAC value of zero implies there were no good hits on that side of the BBC.

RBT File: bbc_bb101_2009_a.rbt

Users: BB101

Inputs: Ch0/1 = QT Board BB001 (East)
Ch2/3 = QT Board BB002 (West)
Ch4/7 = Unused

From each QT board:
bits 0:15 = ADC-Sum
bits 16:27 = Max TAC (Value of zero implies NO good hits)

LUT: 1:1

Registers:

Four registers, all thresholds can be set independently
R0: BBCsmall-EastADCsum_th (16 bits)
R1: BBCsmall-WestADCsum_th (16)
R2: BBCsmall-EastTAC-select (3)
0 => select bits 0:6
1 => select bits 1:7
…
5 => select bits 5:11
R3: BBCsmall-WestTAC-select (3)
Same value definitions as for R2

Action:

1st Latch input

2nd Compare each ADC-sum to its threshold
Calculate: TAC difference = 4096 + TAC-E – TAC-W
Define: Good-TAC-E = TAC-E > 0, same for West side
Make all possible bit selections from TAC-E and TAC-W, including overflow logic. For example:
TAC-E-overflow-0 = TAC-E(7), (8), (9), (10) or (11)
If (TAC-E-overflow-0 = 1) then TAC-E-scaler-0 = 127
Else TAC-E-scaler-0 = TAC-E(0:6)
Same logic for all possible bit selections from TAC-E (see description of register R2) and TAC-W

3rd Delay ADC-sum threshold bits
Zero out TAC difference if either Good-TAC-E or Good-TAC-W is false, otherwise just delay TAC difference
Use R2 to select the TAC-E scaler bits:
If (R2 = 0) then chose TAC-E-scaler-0
Else if (R2 = 1) then chose TAC-E-scaler 1
Etc…
Do the same for West side, using R3 to control the selection.

4th Latch output

Output to VT201:

(0-12) TAC difference
(13) Unused
(14) ADC-sum-E > th0
(15) ADC-sum-W > th0

Scalers:

(0-6) selected bits of TAC-E
(7-13) selected bits of TAC-W
(14) ADC-sum-E > th0
(15) ADC-sum-W > th0

2. Layer 0 QT Boards: BBQ_BB003

There is just one BBC large-tile QT board and it receives data from both the East and West sides of the detector. See documentation from Chris Perkins for a description of this algorithm.

3. Layer 1 DSM Board: BBC_BB102

The BB102 DSM board processes data from the BBC-large-tile detector. The algorithm receives a hit flag and fastest-TAC data for each of the East and West sides of the detector from the QT board. The hit flags indicate there was at least one good hit on each side, and they are just passed through to the output. A set of bits specified by the user is chosen from each incoming TAC value to send to the scaler system. In parallel, the TAC difference is calculated. The difference is set to zero if either of the two incoming TACS is zero, because a TAC value of zero implies there were no good hits on that side of the BBC.

RBT File: bbc_bb102_2010_b.rbt

Users: BB102

Inputs: Ch0/1 = QT Board BB003 (East and West)
Ch2/7 = Unused

From the QT board:
bits 0:11 = MAX TAC East (value of zero implies no good hits)
bits 12:23 = MAX TAC West
bit 24 = East hit
bit 25 = West hit

LUT: 1:1

Registers:

R0: BBClarge-EastTAC-select (3)
0 => select bits 0:6
1 => select bits 1:7
…

5 => select bits 5:11

R1: BBClarge-WestTAC-select (3)

Same value definitions as for R0

Action:

1st Latch input

2nd Delay hit bits to 4th step
Calculate: TAC difference = 4096 + TAC-E – TAC-W
Define: Good-TAC-E = TAC-E > 0, same for West side
Make all possible bit selections from TAC-E and TAC-W, including overflow logic. For example:
TAC-E-overflow-0 = TAC-E(7), (8), (9), (10) or (11)
If (TAC-E-overflow-0 = 1) then TAC-E-scaler-0 = 127
Else TAC-E-scaler-0 = TAC-E(0:6)
Same logic for all possible bit selections from TAC-E (see description of register R0) and TAC-W (see register R1)

3rd Zero out TAC difference if either Good-TAC-E or Good-TAC-W is false, otherwise just delay TAC difference to the 4th step
Use R0 to select the TAC-E scaler bits:
If (R0 = 0) then chose TAC-E-scaler-0
Else if (R0 = 1) then chose TAC-E-scaler-1
Etc…
Do the same for West side, using R1 to control the selection.

4th Latch output

Output to VT201:

(0-12) TAC difference
(13) Unused
(14) East hit
(15) West hit

Scalers:

(0-6) selected bits of TAC-E
(7-13) selected bits of TAC-W
(14) East hit
(15) West hit

4. Layer 0 QT Boards: BBQ_VP001:002

The two VPD QT boards use the same algorithm as is used by the two small-tile BBC QT boards. Please see documentation from Chris Perkins for a description of this algorithm.

5. Layer 1 DSM Board: BBC_VP101

RBT File: bbc_vp101_2009_a.rbt

Users: VP101

Inputs: Ch0/3 = Unused
Ch4/5 = QT Board VP003 (East)
Ch6/7 = QT Board VP004 (West)

The VP101 DSM board receives VPD data from 2 QT boards. The logic needed to do this analysis is the same as that used by the BB101 algorithm. The VP101 algorithm is therefore identical to the BB101 algorithm in every way, except for the input map. Please see the BBC_BB101 documentation above for details of the logic.

6. Layer 0 QT Board: BBQ_ZD001

For the2011 19 GeV and 200 GeV AuAu data taking we have reverted to the 2010 version of the ZDC QT algorithm, which was originally created for heavy-ion running. Please see documentation provided by Chris Perkins for a description.

7. Layer 1 DSM Board: BBC_ZD101

The ZD101 DSM board processes data from the ZDC detector. The algorithm receives fastest-TAC data from the QT boards. A set of bits specified by the user is chosen from each incoming TAC value to send to the scaler system. This is a variation on the logic used in the BB101 algorithm. The difference is that a smaller set of bits is available to send to the scaler system, so the user has a larger number of subsets to choose from. In parallel, the TAC difference is calculated. The difference is set to zero if either of the two incoming TACS is zero, because a TAC value of zero implies there were no good hits on that side of the ZDC. A user-specified set of bits is then chosen to be passed on to VT201. In parallel, the algorithm also receives the results of comparing sums to thresholds. Most of those threshold bits are passed through to VT201 unmodified. Two thresholds, one from each side, are gated with a deadtime signal. This allows the user to zero out those two thresholds for a certain number of RHIC clock ticks after a ZDC coincidence is detected.

RBT File: bbc_zd101_2010_a.rbt

Users: ZD101

Inputs: Ch0/1 = QT Board ZD001
Ch2:7 = Unused

From the QT board:
bits 0:11 = Max TAC, West
bits 12:23 = Max TAC, East
bit 24 = West analog sum > threshold-0
bit 25 = West analog sum > threshold-1
bit 26 = West analog sum > threshold-2
bit 27 = East analog sum > threshold-0
bit 28 = East analog sum > threshold-1
bit 29 = East analog sum > threshold-2
bit 30 = East+West attenuated analog sum > threshold
bit 31 = Unused

LUT: 1:1

Registers:

R0: ZDC-TACdiff-select (2 bits)

0 => select bits 0:8
1 => select bits 1:9
2 => select bits 2:10
3 => select bits 3:11
4 => select bits 4:12

R1: ZDC-EastTAC-select (3)

0 => select bits 0:4
1 => select bits 1:5
…
7 => select bits 7:11

R2: ZDC-WestTAC-select (3)

Same value definitions as for R1

R3: ZDC-deadtime (4 bits)

Action:

1st Latch input

2nd Delay threshold bits to the 4th step.
Gate a copy of the threshold-0 bits with the "take_next" bit, i.e.
Gated-West-th0 = West-th0 and take_next
Gated-East-th0 = East-th0 and take next
Calculate: TAC difference = 4096 + TAC-E – TAC-W
Define: Good-TAC-E = TAC-E > 0, same for West side
Make all possible bit selections from TAC-E and TAC-W, including overflow logic. For example:
TAC-E-overflow-0 = TAC-E(5), (6), (7), (8), (9), (10) or (11)
If (TAC-E-overflow-0 = 1) then TAC-E-scaler-0 = 31
Else TAC-E-scaler-0 = TAC-E(0:4)
Same logic for all possible bit selections from TAC-E (see description of register R1) and TAC-W

3rd Delay the gated threshold bits to the 4th step.
Use R0 to select the TAC difference bits for VT201, including overflow logic and the "good" TAC cut, i.e.:
Diff-overflow-0 = TAC-diff(9), (10), (11) or (12)
Diff-overflow-1 = TAC-diff(10), (11) or (12)
Diff-overflow-2 = TAC-diff(11), (12)
Diff-overflow-3 = TAC-diff(12)
If (Good-TAC-E = 0 or Good-TAC-W = 0) then output = 0
Else if (R0 = 0)
If (Diff-overflow-0 = 1) then output = 511
Else output = TAC-diff(0:8)
Else if (R0 = 1)
If (Diff-overflow-1 = 1) then output = 511
Else output = TAC-diff(1:9)
Etc…

Use R1 to select the TAC-E scaler bits:
If (R1 = 0) then chose TAC-E-scaler-0
Else if (R1 = 1) then chose TAC-E-scaler-1
Etc…
Do the same for the West side using R2 to control the selection.
Check for a ZDC coincidence:
Coincidence = Gated-West-th0 and Gated-East-th0 and
Good-TAC-W and Good-TAC-E
If there is a coincidence then initialize a counter to R3-1. Allow it to count down to zero at a rate of one count per tick of the RHIC clock. Set the "take_next" bit to zero while the counter is counting.

4th Latch output

Output to VT201:

(0-8) TAC difference
(9) Gated-sum-W > th0
(10) Analog-sum-W > th1
(11) Analog-sum-W > th2
(12) Gated-sum-E > th0
(13) Analog-sum-E > th1
(14) Analog-sum-E > th2
(15) Attenuated-sum-E+W > th

Scalers:

(0-4) selected bits of TAC-E
(5-9) selected bits of TAC-W
(10) Gated-sum-W > th0
(11) Analog-sum-W > th0
(12) Gated-sum-E > th0
(13) Analog-sum-E > th0
(14) ZDC coincidence
(15) Attenuated-sum-E+W >th

8. Layer 2 Vertex DSM Board: L1-VT201

All threshold bits of the Vertex tree from the large and small-tile BBC, the ZDC and the VPD are brought into the Vertex DSM. They are passed on to the TCU, some as individual bits and some in combinations. In parallel all four TAC differences are brought into the Vertex DSM. Windows are placed around each TAC difference, and the "inside window" bits get passed through to the TCU and the scaler system. There are two windows around each of the BBC small and large tile TAC differences, but just one window for ZDC and VPD. Some of the TAC difference bits from the BBC small-tiles and the VPD are also in the scaler output. A minimum bias bit, based on an OR of information from all 4 detectors is created. This bit is used to start a counter whose status can be used to provide preceded protection for subsequent triggers. Finally, the atomcules logic is also implemented here.

RBT File: l1_vt201_2011_b.rbt

Users: VT201

Inputs: Ch 0 = BB101
Ch 1 = BB102
Ch 2 = ZD101
Ch 3 = Unused
Ch 4 = VP101
Ch 5:7 = Unused

From Small tile BBC-DSM BB101
(0-12) Small tile TAC-Difference
(13) Unused
(14/15) Small tile ADC East/West sum > th0

From Large tile BBC-DSM BB102
(0-12) Large tile TAC-Difference
(13) Unused
(14/15) East/West hit

From ZDC DSM ZD101
(0-8) ZDC TAC-Difference
(9) ZDC West gated analog sum > th0
(10) ZDC West analog sum > th1
(11) ZDC West analog sum > th2
(12) ZDC East gated analog sum > th0
(13) ZDC East analog sum > th1
(14) ZDC East analog sum > th0=2
(15) ZDC East+West attenuated sum > th0

From VPD-DSM VP101
(0-12) VPD TAC-Difference
(13) Unused
(14/15) VPD ADC East/West> th0

LUT: Either 1-to-1 or TAC-difference range conversion

Registers:
R0: BBCsmall-TACdiff-Min (13 bits)
R1: BBCsmall-TACdiff-Max (13)
R2: BBClarge-TACdiff-Min (13)
R3: BBClarge-TACdiff-Max (13)
R4: ZDC-TACdiff-Min (9)
R5: ZDC-TACdiff-Max (9)
R6: VPD-TACdiff-Min (13)
R7: VPD-TACdiff-Max (13)
R8: Minimum-Bias-Select (4)
R9: Atomcules-Central-Select (5)
R10: Atomcules-Waiting-Int-Select (4)
R11: Atomcules-Triggering-Int-Select (4)
R12: Atomcules-Wating-Time (5)
R13: Atomcules-Trigger-Time (5)
R14: BBCsmall-TACdiff-Window2-Min (13)
R15: BBCsmall-TACdiff-Window2-Max (13)

R16: BBClarge-TACdiff-Window2-Min (13)
R17: BBClarge-TACdiff-Window2-Max (13)
R18: Min_Bias_Protection_Time (9)

Action

1st Latch inputs

2nd Delay the threshold bits from BBC-small, BBC-large, and VPD to the 3rd step along with ZDC th0 bits and ZDC attenuated sum bit.
Delay a second copy of all of the BBC-small and BBC-large bits to the 4th step.
Combine the ZDC th1 and th2 bits to make windows on the East and West sides separately, i.e.:

ZDC-East-Window = ZDC-E>th1 and not ZDC-E>th2
ZDC-West-Window = ZDC-W>th1 and not ZDC-W>th2

Combine the threshold bits from BBC-small, BBC-large, ZDC and VPD to make the atomcules waiting-stage interaction bit. Use R10 to turn each component on/off, i.e.:

Waiting_int = (R10(0) and (BBC-S-E>th0 or BBC-S-W>th0)) or
(R10(1) and (BBC-L-E>th0 or BBC-L-W>th0)) or
(R10(2) and (ZDC-E>th0 or ZDC-W>th0)) or
(R10(3) and (VPD-E>th0 or VPD-W>th0))

Combine the threshold bits from BBC-small, BBC-large, ZDC and VPD to make the atomcules triggering-stage interaction bit. Use R11 to turn each component on/off, i.e.:

Triggering_int= (R11(0) and (BBC-S-E>th0 or BBC-S-W>th0)) or
(R11(1) and (BBC-L-E>th0 or BBC-L-W>th0)) or
(R11(2) and (ZDC-E>th0 or ZDC-W>th0)) or
(R11(3) and (VPD-E>th0 or VPD-W>th0))

Check each of R12 and R13 to see if they are non-zero. If either is zero then the appropriate stage of the atomcules logic is disabled.
Delay a copy of the BBC-small and VPD TAC difference to the 4th step.
Compare each of the 4 TAC differences to its minimum and maximum value, as specified in the relevant registers. The logic looks for the TAC difference to be greater than the minimum and less than the maximum. This process is done twice, in parallel, for each of the BBC-small and BBC-large TAC differences, since they are compared to two sets of minimum and maximum values.

3rd Combine (AND) the ZDC th0 bits to make the ZDC coincidence bit
Combine (AND) the two ZDC window bits to make the ZDC UPC bit
Combine (AND) the VPD threshold bits to make the VPD coincidence bit
Combine the results of the TAC difference comparisons to determine if each TAC difference is inside its specified window, e.g.:

ZDC-Tdiff = R4 < ZDC TAC difference < R5

Combine the results of the TAC difference comparisons and the ADC threshold bits to make the minimum bias bit, using R8 to turn each component on/off, i.e.:

MB = (R8(0) and BBC-S-Tdiff and BBC-S-E>th0 and BBC-S-W>th0) or
(R8(1) and BBC-L-Tdiff and BBC-L-E>th0 and BBC-L-W>th0) or
(R8(2) and ZDC-Tdiff) or
(R8(3) and VPD-Tdiff)

The preceded logic is only enabled if R18 is set to a non-zero value. In this case, whenever the minimum bias bit is set a counter is initialized to R18-1. The counter then counts down to zero at a rate of one count per tick of the RHIC clock. If another minimum bias interaction occurs while the counter is counting, then the counter is re-initialized to R18-1 and counting continues. The preceded bit is true whenever the current counter value is non-zero, and false otherwise.

Combine the results of the TAC difference comparisons with the threshold bits to make the atomcules central bit, Use R9 to turn each component on/off, i.e.:

Cent = ((R9(0) and BBC-S-E>th0 and BBC-S-W>th0 and BBC-Tdiff) or not R9(0)) and
((R9(1) and VPD-E>th0 and VPD-W>th0 and VPD-Tdiff) or not R9(1)) and
((R9(2) and ZDC-E>th0 and ZDC-W>th0 and ZDC-Tdiff) or not R9(2)) and
((R9(3) and not ZDC-E+W>th) or not R9(3)) and
((R9(4) and BBC-L-E>th0 and BBC-L-W>th0) or not R9(4))

NOTE: the second set of TAC difference comparisons for BBC-small and BBC-large are NOT used in the definition of the atomcules central bit.

If there is a central interaction, and R12 is non-zero, then initialize the Waiting-Counter to R12-1. Allow it to count down to zero at a rate of one count per tick of the RHIC clock. Stop counting if a waiting interaction (Waiting_int) occurs while counting is in progress.
If the counter reaches zero without being stopped, and R13 is non-zero, then initialize the Triggering-Counter to R13-1. Allow it to count down to zero at a rate of one count per tick of the RHIC clock. Stop counting if a triggering interaction (Triggering_int) occurs while this second stage of counting is in progress.
The Atomcules output bit is set while the Triggering-Counter is counting.
NOTE: If R12 is zero then the waiting stage is disabled so the triggering stage never gets initiated.

4^{th} Latch Outputs

Output to TCU:

Bit	Name	Description
Bit 0	BBC-TAC	BBC small-tile TAC difference in window
Bit 1	BBC-E	BBC small-tile East ADC sum > threshold
Bit 2	BBC-W	BBC small-tile West ADC sum > threshold
Bit 3	BBC-L-TAC	BBC large-tile TAC difference in window
Bit 4	BBC-L-E	BBC large-tile East hit
Bit 5	BBC-L-W	BBC large-tile West hit
Bit 6	ZDC-TAC	ZDC TAC difference in window
Bit 7	BBC-TAC2	BBC small-tile TAC difference in window 2
Bit 8	BBC-L-TAC2	BBC large-tile TAC difference in window 2
Bit 9	ZDC-UPC	ZDC East analog sum in window AND ZDC West analog sum in window
Bit 10	ZDC-COINC	ZDC East gated analog sum > threshold AND ZDC West gated analog sum > threshold
Bit 11	Minimum-Bias	At least one selected minimum bias component satisfied.
Bit 12	Atomcules	Central collision followed by a period of no interactions
Bit 13	VPD-TAC	VPD TAC difference in window
Bit 14	VPD-COINC	VPD East ADC sum > threshold AND VPD West ADC sum > threshold
Bit 15	Preceded	Counter started by Minimum Bias bit still counting.

Output to Scalers

Bit	Description
Bit 0	BBC small-tile TAC difference in window
Bits 1:4	4 MSB of BBC small-tile TAC difference
Bit 5	BBC large-tile TAC difference in window
Bit 6	ZDC TAC difference in window
Bit 7	ZDC East gated analog sum > threshold AND ZDC West gated analog sum > threshold
Bits 8	Minimum Bias
Bits 9	BBC small-tile TAC difference in window 2
Bits 10	BBC large-tile TAC difference in window 2
Bit 11	VPD TAC difference in window
Bits 12:14	3 MSB of VPD TAC difference
Bit 15	Minimum Bias 2